Web User-session Inference by Means of Clustering Techniques

Web User-Session Inference by Means of Clustering Techniques

Andrea Bianco, *Member, IEEE*, Gianluca Mardente, Marco Mellia, *Member, IEEE*, Maurizio Munafò, *Member, IEEE*, and Luca Muscariello, *Member, IEEE*

***Abstract*—This paper focuses on the definition and identification of "Web user-sessions", aggregations of several TCP connections generated by the same source host. The identification of a user-session is non trivial. Traditional approaches rely on threshold based mechanisms. However, these techniques are very sensitive to the value chosen for the threshold, which may be difficult to set correctly. By applying clustering techniques, we define a novel methodology to identify Web user-sessions without requiring an *a priori* definition of threshold values. We define a clustering based approach, we discuss pros and cons of this approach, and we apply it to real traffic traces. The proposed methodology is applied to artificially generated traces to evaluate its benefits against traditional threshold based approaches. We also analyze the characteristics of user-sessions extracted by the clustering methodology from real traces and study their statistical properties. Web user-sessions tend to be Poisson, but correlation may arise during periods of network/hosts anomalous behavior.**

***Index Terms*—Clustering methods, traffic measurement, web traffic characterization.**

I. Introduction

THE study of telecommunication networks has been often based on traffic measurements, which are used to create traffic models and obtain performance estimates. While a lot of attention has been traditionally devoted to traffic characterization at the packet and transport layers (see for example [1]–[6]), few are the studies on traffic properties at the session/user layer [1], [7], [8]. This is due to the difficulty in defining the "session" concept itself [9], which depends on the considered application. Applications such as *telnet* or *ssh* typically generate a single TCP connection per single user-session, whereas application layer protocols such as HTTP, IMAP/SMTP and X11 usually generate multiple TCP connections per user-session. Also, the generally accepted conjecture that such sessions follow a Poisson arrival process (see [10] for example) might have reduced the interest in the user-session process analysis.

User-session identification and characterization play an important role both in Internet traffic modeling and in the proper dimensioning of network resources. Besides increasing the knowledge of network traffic and user behavior, they yield workload models which may be exploited for both performance evaluation and dimensioning of network elements. Synthetic workload generators may be defined to assess network performance, e.g., benchmarking of server farms, firewalls, proxies or NATs, as in [11], [12]. Similarly, user-session characterization allows researchers to build realistic scenarios when assessing the performance of a complex network via simulation. Furthermore, network dimensioning problems are usually based on simple assumptions to permit analytical formulations and solutions. The validation of these assumptions can only be obtained by checking the model against traffic measurements. Finally, the knowledge of user-session behavior is important for service providers, for example to dimension access links and router capacity. User behavior can be modeled by few parameters [13]–[15], e.g., session arrival rates, data volumes. Operators are interested in monitoring these parameters, especially today that traffic demands change very quickly as new services are continuously proposed to customers. Thus, correct user-session identification and characterization are of fundamental importance and interest.

A Web user-session, simply named user-session in the remainder of the paper, is informally defined as a set of TCP connections created by a given user while surfing the Web during a given time frame. The main paper goals are: (i) to devise a technique that permits to correctly identify user-sessions, and, (ii) to determine their statistical properties by analyzing traces of measured data. The described methodology can be easily extended, for example, to identify either P2P lookup sessions, i.e., groups of subsequent queries, or TCP connections that can be grouped to define an SMTP "mail session". However, since the Web is the most widely used interactive service, we concentrate on the identification of Web user-sessions generated by a single host.

We assume that a single user runs a browser on each host, a reasonable hypothesis today given the vast majority of PC based hosts. A user-session informal definition can be obtained by describing the typical behavior of a user browsing the Web. An activity (ON) period on the Web alternates with a silent (OFF) period during which the user is inactive on the Internet. This activity period, named session in this paper, may comprise several TCP connections, opened by the same host toward possibly different servers.

Clustering techniques [18] are exploratory techniques used in many areas to analyze large data sets. Given a proper notion of similarity, they find groups of similar variables/objects by partitioning the data set in "similar subsets". Typically, several metrics over which a distance measure can be defined are

Manuscript received November 10, 2006; revised June 12, 2007, and December 21, 2007; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor H. Schulzrinne. First published July 25, 2008; current version published April 15, 2009.

A. Bianco, M. Mellia, and M. Munafò are with the Dipartimento di Elettronica, Politecnico di Torino, Torino 10129, Italy (e-mail: andrea.bianco@polito.it; marco.mellia@polito.it; maurizio.munafo@polito.it).

G. Mardente is with Cisco Systems, San Jose, CA 95134 USA (e-mail: mgianluc@cisco.com).

L. Muscariello is with France Telecom R&D, Issy-Les-Moulineaux 92794, France (e-mail: luca.muscariello@orange-ftgroup.com).

Digital Object Identifier 10.1109/TNET.2008.927009

associated with points (named samples) in the data set. Informally, the partitioning process tries to put neighboring samples in the same subset and distant samples in different subsets. The aim of this paper is to define a clustering technique to identify user-sessions. Performance is compared with those of traditional threshold based approaches, which partition samples depending on a comparison between the sample to sample distance and a given threshold value. The main advantage of the clustering approach is avoiding the need to define *a priori* any threshold value to separate and group samples. Thus, this methodology is more robust than simpler threshold based mechanisms.

The main contributions of this paper are the following. First, we adapt classical clustering techniques to the described scenario, a nontrivial task that requires ingenuity to optimize the performance of user-session identification algorithms both in terms of speed and precision. By running a clustering algorithm, we avoid the need of setting *a priori* a threshold value, since clustering techniques automatically adapt to the actual user behavior, as better explained later. Furthermore, the algorithm does not require any training phase to properly run. We test the proposed methodology on artificially generated traces i) to ensure its ability to correctly identify a set of TCP connections belonging to the same user-session, ii) to assess the error performance of the proposed technique, and iii) to compare it with traditional threshold based mechanisms. Analytical results are presented to determine the performance of threshold based mechanisms. Finally, we run the algorithms over real traffic traces, to obtain statistical information on user-sessions, such as distributions of i) session duration, ii) amount of data transferred in a single session, iii) number of connections within a single session. A study of the inter-arrival times of Web user-sessions is also presented, from which it emerges that Web user-sessions tend to be Poisson, but correlation may arise due to network/hosts anomalous behavior. Preliminary results on user-sessions statistical characterization were presented in [19].

II. Related Work

The definition of a user-session is not straightforward. As previously mentioned, a common definition of a user-session is given by a period of time during which the user is generating traffic. A user-session is then terminated by a "long" inactivity period. Within an activity period, many TCP connections may be used to transfer data. Unfortunately, the identification of active and silent periods is not trivial and the definition of the user activity may also depend on the selected application.

To the best of our knowledge, the first attempt to consider the "session" arrival process was presented in [1]. However, the focus is on Telnet and FTP sessions, where each session is related to a single TCP data connection. No measurements of HTTP sessions are reported.

To identify HTTP user-sessions, traditional approaches rely on the adoption of a threshold η [7], [8]. TCP connections are aggregated in the same session if the inter-arrival time between two TCP connections is smaller than the threshold value. Otherwise, the TCP connection is associated with a newly created user-session. In [7], η is selected to be 100 s, while in [8] a threshold $\eta = 1$ s is chosen. Results are obviously affected by the choice of η. Indeed, the threshold-based approach works well only if the threshold value is correctly matched to the values of connection and session inter-arrival times. Furthermore, different users may show different idle times, and even the same user may have different idle periods depending on the service, e.g., news or e-commerce, he is accessing. Thus, the *a priori* knowledge of the proper threshold value is an unrealistic assumption. If the threshold value is not correctly matched to the session statistical behavior, threshold based mechanisms are significantly error prone, as we will show in Section VI. To avoid this drawback, we propose a more robust algorithm to identify user-session.

When considering Web users' characterization, many authors perform a data analysis of server logs to define user-sessions (see [16] for example). While the server log approach can be very reliable, it lacks the capability offered by passive measurements performed at the packet level which permit to simultaneously monitor a user browsing several servers. To this extent, authors in [17] adopt a passive sniffing methodology to rebuild HTTP layer transactions to infer clients/users' behaviors. By crawling HTTP protocol headers, the sequence of objects referred by the initial request is rebuilt. This allows grouping several TCP connections to form a user-session. In this paper, we focus on user behavior measured at the client side, and not at the server side. In this scenario, the server log analysis approach is impractical in assessing the behavior of *users* at the access network, since user-sessions include connections to several different servers.

While the server log approach can be very effective, it does not scale well and, by leveraging on a specific application level protocol, can be hardly generalized. Furthermore, since the payload of all packets must be analyzed, this approach is not practical when, for security or privacy reasons, data payload (and application layer headers) are not available. Thus, in this paper, TCP headers only are analyzed, limiting privacy issues and significantly reducing the probe complexity. Furthermore, the proposed approach may be adopted for any set of sessions generated by the same application, even to sessions collected by analyzing Web server logs. Our methodology is rather general, and is much more robust than any threshold based approach.

III. Clustering Techniques

In this section we briefly describe the basics of clustering techniques to provide an overview of their main features. More details on clustering techniques can be found in [18]. Informally, clustering algorithms group objects that have similar characteristics in clusters, according to a notion of distance among objects. Our goal is to exploit this property to group connections (objects) to identify user-sessions (clusters) in an automatic fashion.

Let us consider a metric space X, named sampling space, and a set of samples $A = \{x_1, \ldots, x_N | x_i \in X\}$ which have to be grouped (clustered) into K subsets: we wish to find a partition $\mathcal{C} = \{C_1, \ldots, C_K\}$, such that $\bigcup_i C_i = A$ and $C_i \cap C_{j \neq i} = \emptyset$, with K possibly being unknown *a priori*. The subsets in the partition are named *clusters*. Clusters contain "similar" samples, whereas samples associated with different clusters should be "dissimilar", the similarity being measured via the

sample-to-sample and cluster-to-cluster distances. Depending on the data set, an ad hoc distance definition should be provided on the basis of a trial and error procedure. Let us assume for simplicity that $X = \mathbb{R}^n$, where n is the sampling space dimension. x_i^k represents the kth component of sample x_i, the sample distance $d(x_i, x_j) = \sqrt{\sum_{k=1}^{n}(x_i^k - x_j^k)^2}$ is the classical Euclidean metric, and the distance between two clusters C_i,C_j is defined as

$$d(C_i, C_j) = \min_{x \in \mathcal{R}(C_i), y \in \mathcal{R}(C_j)} d(x, y) \tag{1}$$

where $\mathcal{R}(C)$ is a set of selected points representing the whole cluster C.

Now we briefly introduce two clustering techniques, known in the data mining and in the descriptive multivariate statistic environments as "hierarchical agglomerative" and "partitional" clustering.

A. The Hierarchical Agglomerative Approach

Each sample is initially associated with a different cluster, i.e., $C_i = \{x_i\}$; thus, at procedure startup the number of clusters is $N_c = |A|$. Then, on the basis of the definition of a cluster-to-cluster distance, the clusters at minimum distance are merged to form a new cluster. The algorithm iterates this step until all samples belong to the same cluster $C = A = \{x_1, \ldots, x_N\}$, and $N_c = 1$.

This procedure defines a merging sequence based on minimum distance between clusters. At each step $s = 1, \ldots, N$, a *quality indicator* function $\gamma^{(s)}$ is evaluated. The set A is finally clustered by selecting the number of clusters $N_c = N - (s-1)$ such that $\gamma^{(s)} - \gamma^{(s-1)}$ is maximized. Intuitively, the quality indicator function $\gamma^{(s)}$ measures the distance between the two closest clusters at step s. A sharp increase in the value of $\gamma^{(s)}$ is an indication that the merging procedure is merging two clusters which are too far apart, thus suggesting to adopt the previous partition as the best cluster configuration.

This approach can be quite time consuming, especially when the data set is very large, since the initial number of clusters N_c is equal to the number of samples in the data set $|A|$. For this reason, non-hierarchical approaches, named partitional, are often preferred, since they show better scalability properties.

B. The Partitional Approach

This technique is used when the final number of clusters K is known. The procedure starts with an initial configuration including K clusters, selected according to some criteria. The final cluster definition is obtained through an iterative procedure.

The cluster C_i is represented by a subset of samples $\mathcal{R}(C_i)$ when measuring cluster-to-cluster distance. For example, the algorithm is named K-means algorithm when the cluster representative is the so-called *centroid* $\hat{c}_i$, defined as the mean value of the cluster samples, i.e.,

$$\hat{c}_i^k = \frac{1}{|C_i|} \sum_{x \in C_i} x^k, \quad k = 1, \ldots, n,$$

where n is the size of the sampling space.

At procedure startup, K clusters are created, with cluster centroids selected according to a given rule in the measurement space, e.g., to partition the measurement space in $K + 1$ equispaced areas or randomly. Each sample is associated with the closest cluster, according to the distance between the sample and the centroid of each cluster. When all samples are assigned to a cluster, new centroids are computed and the procedure iterates. The algorithm ends when either a prefixed number of iterations is reached, or the number of samples which are moved to a different cluster is negligible according to a predefined threshold. The final result may change depending on the chosen initial state.

IV. Using Clustering Techniques on the Measurement Data Set

In this section we describe the methodology developed for the identification of Web user-sessions, including the choices made in the cluster analysis. The description refers to the analysis of data collected through measurements; the same process is also used to analyze artificial traffic, when trying to determine the procedure ability to correctly identify known user-sessions. We start by giving some details about the analyzed data set of traces, to define the variables that will be used by the clustering algorithm.

A. Traffic Trace Description

Traffic traces were collected on the Internet access link of Politecnico di Torino, i.e., between the border router of Politecnico and the access router of GARR/B-TEN [20], the Italian and European Research network. The Politecnico campus LAN comprises roughly 7 000 hosts; most of them are clients, but several servers are also regularly accessed from outside. The campus LAN backbone is based on a switched Fast Ethernet infrastructure, which behaves as a stub Internet subnetwork; indeed, there is a single 28 Mbps point of access to the GARR/B-TEN network and, through it, to the public Internet. A strict regulation of the network facilities is imposed by a firewall which blocks (most of) the peer-to-peer traffic. Thus, still today, the majority of Politecnico Internet traffic is created by Web browsing. Details on the measurement setup and traffic characteristics are available in [21] and [22].

Since 2001, several traces have been regularly collected. Among the available data, we selected two different periods:

- Oct. 02: from 23/10/2002 to 31/10/2002;
- Apr. 04: from 29/4/2004 to 6/5/2004.

Both periods comprise more than a week of continuously traced data. We performed the analysis by considering only working hours, i.e., traffic from 8 AM to 8 PM, Monday to Friday. In Section VII we mainly report results on the Apr. 04 period, if not otherwise stated, being the difference among the two data sets almost negligible. However, the Apr. 04 data set includes two working days (May 5th and 6th) during which hosts in our campus were attacked and infected by the "Sasser.B" worm [23]. The worm infection does not affect our measurement campaign directly, since the worm spreading is not based on the HTTP protocol.

Bidirectional packet level traces were collected using *tcpdump* [24], and later processed by *Tstat* [22] to obtain a TCP

level trace. Among other data, Tstat produces a TCP level log file, which collects all observed TCP connections. A TCP connection is considered opened when the first SYN segment from the client is observed, and it is terminated either when the tear-down sequence is observed (either the FIN/ACK or RST messages), or when no segment is observed for more than 15 minutes.[1] Only TCP connections whose three-way-handshake is successfully completed are tracked; thus, misbehaving connections and activities (e.g., port-scanning) do not affect the data set. We used Tstat to track:

- $f_{\mathrm{id}} = (\mathrm{IP}_c, \mathrm{IP}_s, \mathrm{TCP}_c, \mathrm{TCP}_s)$: the 4-tuple identifying the connection, i.e., IP addresses and TCP port numbers of the client and the server;
- $t(f_{\mathrm{id}})$: the connection opening time, identified by the timestamp of the first client SYN message;
- $t_d(f_{\mathrm{id}})$: the time instant in which the last segment carrying data is observed (either from the client or from the server);
- $t_e(f_{\mathrm{id}})$: the connection ending time, identified by the time instant in which the TCP connection is terminated;
- $B_c(f_{\mathrm{id}})$ and $B_s(f_{\mathrm{id}})$: the net amount of byte sent from the client and server respectively (excluding retransmissions).

B. Clustering Algorithm Description

The first issue is to choose the n statistical variables needed to define the metric space $X = \mathbb{R}^n$ for the clustering analysis; this implies selecting the metric space that best fits our problem. The easiest approach is to let the clustering algorithm run over a large number of statistical variables, typically including the majority of available data (in our example, potential variables may be IP source address, TCP destination port, TCP connection opening and ending time, etc.).

However, after several trials, we discovered that an accurate pre-filtering of captured data improves the algorithmic speed and provides more accurate results. Since we wish to identify a Web user-session, i.e., a group of TCP connections corresponding to the activity period of a user running a Web browser, we use the opening time t of a TCP connection as the only statistical variable for the clustering process ($n = 1$, and $X = \mathbb{R}$), i.e., the clustering uses a uni-dimensional space.

Since a session groups several connections and starts when a connection starts, the connection start time is a good descriptor to identify the session process. Indeed, including TCP connection ending time leads to misleading results, due to the presence of very large data transfers which may span a long period of time. Similarly, considering IP destination addresses is not helpful, since during a user-session several servers with different IP addresses may be contacted.

Before running the clustering algorithm, the traces are preprocessed according to the following rules. Since NAT and proxy are not used in our campus LAN, we assume that a Web user is identified by its client IP address IP_c, and by connections having TCP server port TCP_s equal to 80 (HTTP protocol). To consider only significant IP addresses (users), we selected the most active hosts among the about 7000 IP addresses that appear in the traces, i.e., the 1500 campus LAN IP addresses with the larger number of generated TCP connections. Each user trace, i.e., a trace containing only data with a given IP source address, is preprocessed according to the following steps: i) data are partitioned day by day, ii) only working hours of working days are considered, and iii) opening times of two consecutive connections separated by more than half an hour are considered *a priori* as two independent data sets. Thus, for a given host with IP address IP^*, the set $A(\mathrm{IP})$, whose elements are TCP connection opening times within a given time-frame, represents the samples of the clustering procedure:

$$A(\mathrm{IP}) = \{t(f_{\mathrm{id}}) | \mathrm{IP}_c = \mathrm{IP}^*, \mathrm{TCP}_s = 80, t_i(f_{\mathrm{id}}) - t_{i+1}(f_{\mathrm{id}}) < 1800\ \mathrm{s}\}$$

where $t_i(f_{\mathrm{id}})$ is the connection opening time associated with sample i.

The pre-processing steps allow the clustering algorithm to concentrate only on TCP connections created by a single user: very long silent periods do not interfere with the user session identification process. Furthermore, since the Web traffic at night is negligible and any dimensioning algorithm would be interested in looking at peak hour behavior, the trace are split on a daily basis to limit the data set the clustering algorithm has to process and to reduce the processing run time.

After the metric space definition, a proper cluster analysis methodology must be selected. Recall that hierarchical agglomerative cluster analysis proceeds by creating clusters through a cluster merging procedure: this methodology is easy to implement, but it does not scale well with the number of samples, which in our case is fairly large. Indeed, during Oct. 02, $N = |A(\mathrm{IP})|$ ranges up to 57000, while during Apr. 04, $N = |A(\mathrm{IP})|$ ranges only up to 26000, since the spread of the Sasser.B worm forced network administrators to shut off many hosts and subnets. On the other hand, partitional algorithms, which are relatively efficient, require an *a priori* knowledge of the number of clusters K, which is not always easy to predict. Furthermore, in partitional algorithms the representative choice may have a large impact on both the quality of the clustering and the speed of convergence.

To take the advantages and to avoid the drawbacks of both methodologies, we use a mix of them. Thus, for each $A(\mathrm{IP})$, the following three-step algorithm is run to identify user-sessions:

1) an initial clustering is obtained using a partitional algorithm;
2) a hierarchical agglomerative algorithm is used to aggregate the clusters and to obtain a good estimation of the final number of clusters N_c;
3) a partitional algorithm is used to obtain a fine definition of the N_c clusters.

1) Initial Clustering Selection: We start with a partitional algorithm with K clusters, with K significantly smaller than the total number of samples (a study on the impact of K is presented in Section VI). To efficiently position, in our uni-dimensional metric space, the K representatives at procedure startup, we evaluate the distance between any two adjacent samples t_i, t_{i+1}. According to the distance metric $d(t_i, t_{i+1}) = |t_i - t_{i+1}|$, we take the farthest $(K-1)$ couples and determine K intervals. Let $t_{I,\mathrm{inf}}, t_{I,\mathrm{sup}}$ be the inferior and superior bounds of

[1]Given the possibility that a tear-down sequence of a TCP connection under analysis is never observed, a timer is needed to avoid memory consumption. We selected a large value for the timer, according to the findings in [25].

interval I; the centroid position of each cluster is set to $\hat{c}_I = (t_{I,\sup} + t_{I,\inf})/2$, and the partitional algorithm is run for up to 1000 iterations: therefore, K initial clusters are obtained.

Each cluster C is represented by a small subset $\mathcal{R}(C)$ of samples; $|\mathcal{R}(C)| \leq 2$ is enough in our case, since the metric space is $\mathbb{R}$. Possible choices for the representative samples $\mathcal{R}(C)$ are: (i) the cluster centroid, which gives the name "centroid" (or K-means) to the procedure; (ii) the gth and $(100-g)$th percentiles, with $g \neq 0$; (iii) the gth and $(100-g)$th percentiles with $g = 0$, which yields the "single linkage" algorithm.

2) The Hierarchical Agglomerative Procedure: In the second step, a hierarchical agglomerative algorithm is iteratively run, using only the representative samples $\{\mathcal{R}(C)\}$ to evaluate the distance between two clusters. Since the procedure starts with K initial clusters, the number of steps is bounded. At each step s, the hierarchical agglomerative procedure merges the two closest clusters; then, distances among clusters are recomputed. After K iterations, the process ends.

The clustering quality indicator function $\gamma^{(s)}$ permits to select the best clustering among those determined in the iterative process. Indeed, at each step s, the clustering quality must be evaluated to determine if the optimal number of clusters has been found. Denote the jth cluster at step s as $C_j^{(s)}$; at each step, the procedure evaluates the function $\gamma^{(s)}$:

$$\gamma^{(s)} = \frac{d_{\min}^{(s)} - \bar{d}_{\min}^{(s)}}{\bar{d}_{\min}^{(s)}}$$

where

$$d_{\min}^{(s)} = \min_{j,k \neq j} d\left(C_j^{(s)}, C_k^{(s)}\right),$$

$$\bar{d}_{\min}^{(s)} = \frac{1}{s-1} \sum_{l=1}^{s-1} d_{\min}^{(l)},$$

and $d(C_j^{(s)}, C_k^{(s)})$ is defined according to (1).

A sharp increase in the value of $\gamma^{(s)}$ is an indication that the merging procedure is artificially merging two clusters which are too far apart. The optimal number of clusters N_c is determined as

$$N_c = N - \left(\text{argmax}_s(\gamma^{(s)} - \gamma^{(s-1)}) - 1\right)$$

which is computed for the index s that corresponds to the sharpest increase in $\gamma^{(s)}$.

A typical evolution of the function $\gamma^{(s)}$ is reported in Fig. 1, where the sharpest increase is clearly visible. The plot refers to an artificial trace obtained as described in Section VI, and shows that for about 1000 steps the aggregation of the two closest clusters is clearly beneficial in terms of clustering quality. Then, the aggregation process merges two clusters which are too far apart, forcing a sudden increase in $d_{\min}^{(s)}$ at step s, and, therefore, in $\gamma^{(s)}$. When $\gamma^{(s)}$ reaches the maximum, the merging procedure is forcing an artificial aggregation of two distinct clusters. Other errors are induced later in the iterative aggregation process: although clearly visible in Fig. 1, they have a minor impact on the quality indicator function.

3) Final Clustering Creation: A partitional clustering procedure is run over the original data set, which includes all samples,

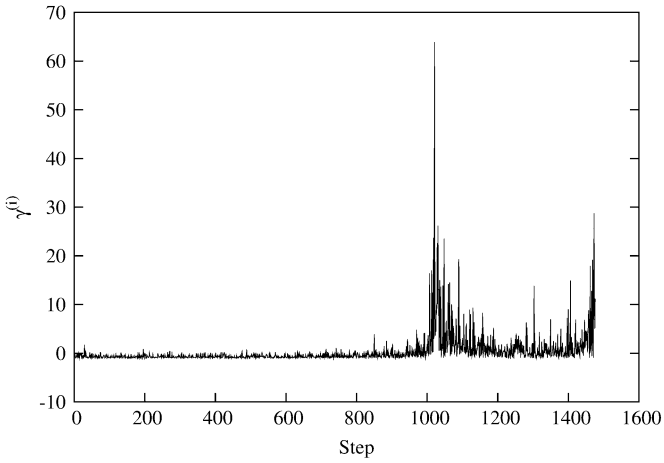

Fig. 1. Sample plot of the quality indicator function $\gamma^{(s)}$.

using the optimal number of clusters N_c determined so far and the same choice of cluster representatives adopted in the first step. A fixed number of iterations is run to obtain a final refinement of the clustering definition. This phase is not strictly required, since at the end of the hierarchical agglomerative procedure a partition is already available. However, it produces clusters of *real* samples instead of representatives (which may not be data points). Furthermore, the computational cost of this phase is almost negligible if compared to the previous one.

V. Modeling Threshold Based System

In this section we model the error probability of threshold based algorithms in detecting user-sessions. Unfortunately, the analytical model for the clustering system proved to be too computationally demanding; as such, it is not useful to obtain performance figures, if not considering too simple scenarios.

Assume X is a random variable; let $\overline{X}$ denote the mean value of X, F_X the Cumulative Distribution Function (CDF) of X, and f_X the Probability Density Function (PDF) of X.

Suppose users generate Web user-sessions according to a stationary renewal ON/OFF process. Let T_{on} denote the time during which a user-session is active. The idle period, denoted by T_{off}, is the time elapsed between the arrival times of the last connection of the current session and of the first connection of the next session. Both T_{on} and T_{off} are assumed to be i.i.d. During the session active period, a user generates connections according to i.i.d. inter-arrivals; let T_{arr} be the random variable describing the connection inter-arrival process. After the last connection within a session has been generated, the session is ended, and the user becomes idle. Let N be the total number of connection inter-arrivals in a data set. We can distinguish among inter-arrivals due to inter-session connections (N_{on}) and intra session connections (N_{off}); thus, $N = N_{\text{on}} + N_{\text{off}}$.

Let us consider a session identification algorithm based on a threshold η. If the connection inter-arrival time is larger than η, two distinct sessions are identified. An error occurs whenever either $T_{\text{arr}} > \eta$ or $T_{\text{off}} < \eta$. Therefore, we can evaluate

$$p_{\text{arr}} = \text{Prob}\{T_{\text{arr}} > \eta\} = 1 - F_{T_{\text{arr}}}(\eta)$$

$$p_{\text{off}} = \text{Prob}\{T_{\text{off}} < \eta\} = F_{T_{\text{off}}}(\eta).$$

Let N_1 and N_2 be the random variables corresponding to the number of errors due to the first and second case respectively. Both variables have a Binomial distribution, $N_1 \sim B(N_{\rm on}, p_{\rm arr})$ and $N_2 \sim B(N_{\rm off}, p_{\rm off})$, because of the i.i.d. property of both session and connection inter-arrival processes. The random variable corresponding to the total number of errors is given by the sum of N_1 and N_2; its distribution is not binomial in general. Let us focus on $\overline{\epsilon}(\eta)$, the mean number of errors normalized over N samples. Since $\overline{N}_{\rm on} = \overline{N}_{\rm off}\overline{T}_{\rm on}/\overline{T}_{\rm arr}$, we have

$$\begin{aligned}\overline{\epsilon}(\eta) &= \frac{\overline{N}_1 + \overline{N}_2}{\overline{N}} = \frac{\overline{N}_1 + \overline{N}_2}{\overline{N}_{\rm on} + \overline{N}_{\rm off}} \\ &= \frac{\overline{N}_{\rm on}\, p_{\rm arr} + \overline{N}_{\rm off}\, p_{\rm off}}{\overline{N}_{\rm off}\overline{T}_{\rm on}/\overline{T}_{\rm arr} + \overline{N}_{\rm off}} \\ &= \rho p_{\rm arr} + (1-\rho) p_{\rm off} \end{aligned} \tag{2}$$

where

$$\rho = \frac{\overline{T}_{\rm on}/\overline{T}_{\rm arr}}{\overline{T}_{\rm on}/\overline{T}_{\rm arr} + 1}.$$

The minimum value for $\epsilon(\eta)$ and the corresponding value of the optimal threshold η^* can be evaluated by solving

$$\frac{\partial \overline{\epsilon}}{\partial \eta} = -\rho f_{T_{\rm arr}}(\eta^*) + (1-\rho) f_{T_{\rm off}}(\eta^*) = 0.$$

Thus,

$$\frac{f_{T_{\rm off}}(\eta^*)}{f_{T_{\rm arr}}(\eta^*)} = \frac{\rho}{1-\rho}. \tag{3}$$

The solution of (3) depends on the assumed distributions. When considering positive random variables with a PDF everywhere convex, the solution is unique.

As an example, consider the case in which all random variables are exponentially distributed, with parameters $\mu_{\rm off} = \overline{T}_{\rm off}/\overline{T}_{\rm arr}$, and $\mu_{\rm on} = \overline{T}_{\rm on}/\overline{T}_{\rm arr}$. By solving (3), the value of the optimal threshold, normalized to $\overline{T}_{\rm arr}$, is

$$\frac{\eta}{\overline{T}_{\rm arr}} = \frac{\mu_{\rm off}}{\mu_{\rm off} - 1} \log \mu_{\rm on}.$$

Fig. 2 reports $\overline{\epsilon}(\eta)$ considering different values of $\overline{T}_{\rm off}$. The percentage of errors grows for decreasing values of η. This is due to the larger probability of misidentifying a connection inter-arrival as a session inter-arrival. Similarly, large values of η lead to session inter-arrivals being identified as connection inter-arrivals. Finally, when $\overline{T}_{\rm off}$ is small (i.e., comparable with $\overline{T}_{\rm on}$), the error probability becomes larger. Similar considerations hold for variable $\overline{T}_{\rm on}$ in Fig. 3: in this case, the range of the optimal threshold values is significantly larger than in the previous plot.

In summary, the presented results show that the threshold based methodology is rather error prone when the optimal value of the threshold is unknown. Furthermore, the optimal value of the threshold is quite variable with respect to the parameters that characterize connection dynamics, making it difficult to use this technique in practice.

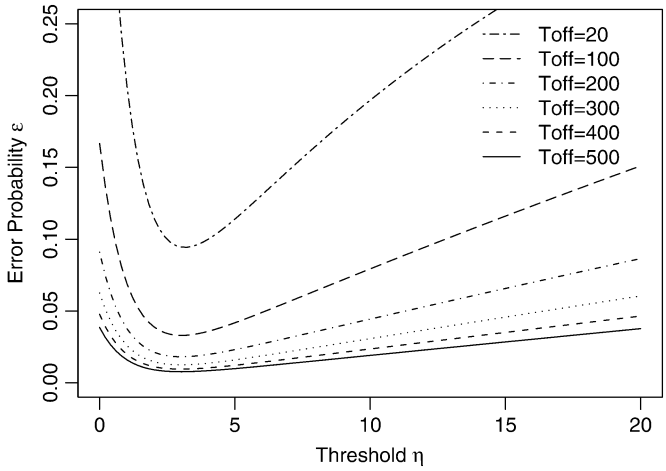

Fig. 2. Error probability as a function of the threshold η for exponential random variables, with $\overline{T}_{\rm on} = 20$ s, $\overline{T}_{\rm arr} = 1$ s, and 20 s $\leq \overline{T}_{\rm off} \leq 500$ s.

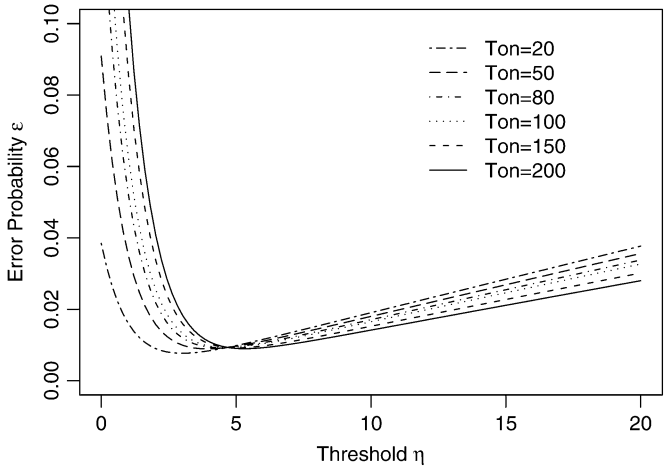

Fig. 3. Error probability as a function of the threshold η for exponential random variables, with $\overline{T}_{\rm off} = 500$ s, $\overline{T}_{\rm arr} = 1$ s, and 20 s $\leq \overline{T}_{\rm on} \leq 200$ s.

VI. Performance Analysis: Artificial Traffic

Let us consider a simple artificial trace in which a *single* user generates sessions according to an ON/OFF process. The session ON and OFF periods are assumed exponentially distributed, with $\overline{T}_{\rm on} = 20$ s, whereas $\overline{T}_{\rm off}$ ranges between 30 s and 2000 s. During each session ON period, a random number of TCP connections is generated, with mean inter-arrival time $\overline{T}_{\rm arr} = 1$ s; we consider both exponential and Pareto distributions.[2] Indeed, the exponential distribution yields a Poisson process, with no possible control on the variance of the connection inter-arrival process ($\sigma^2 = 1$). On the contrary, the Pareto distribution permits to control the process variance. Recall that the Pareto distribution is characterized by two parameters, a and b. The Pareto PDF is

$$f(x) = \frac{ab^a}{x^{a+1}}, \quad x > b$$

[2]We also examined Weibull distributed connection inter-arrivals to consider the case in which the connection inter-arrival process has a variance smaller than 1, but performance results are similar to those shown for exponential and Pareto distributions.

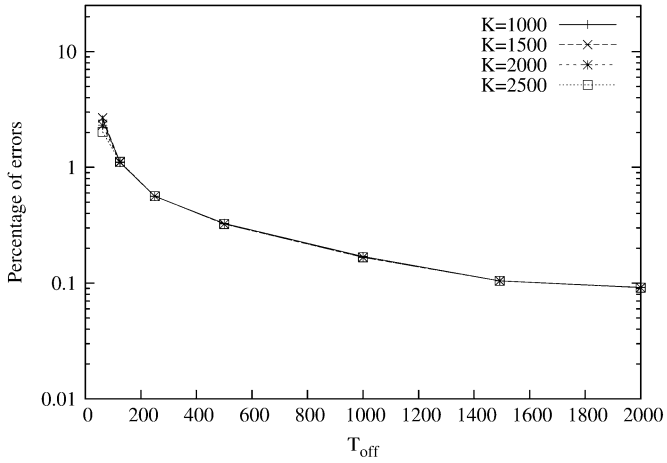

Fig. 4. Clustering sensitivity to the initial number of clusters K for exponential connection inter-arrivals.

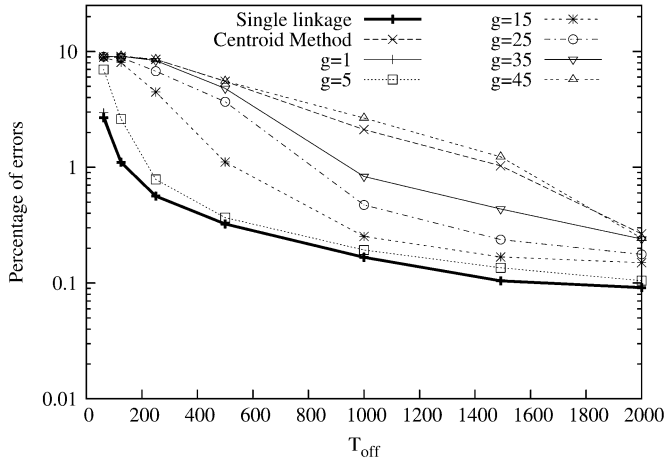

Fig. 5. Clustering sensitivity to the percentile g for exponential connection inter-arrivals.

where the mean μ and variance σ^2 are respectively

$$\mu = \frac{ab}{a-1}$$

$$\sigma^2 = \frac{ab^2}{(a-1)^2(a-2)}.$$

The Pareto distribution shows a heavy-tail, and may have finite or infinite moments of order n depending on the value of a. By selecting $a = 2$, the distribution has finite mean, but infinite variance. This choice makes with high probability the connection inter-arrival time comparable with the session inter-arrival time, therefore creating a demanding scenario. To obtain the mean value $\overline{T}_{\text{arr}} = 1$ s, we set $a = 2$ and $b = 0.5$.

The performance metric is the percentage of misidentified sessions, i.e., the total number of observed errors divided by the total number of connection arrivals times 100. All curves are averaged considering 50 different runs, each comprising 500 sessions (an average of 10 000 connection arrivals per run).

A. Parameter Sensitivity

We initially evaluate the influence on performance results of the values chosen for i) the initial number of clusters K, used in the first clustering phase, and ii) the percentile g, used in the hierarchical agglomerative clustering phase. Considering the exponential connection inter-arrival scenario, in Fig. 4 the error probability is shown to be practically independent from the value of the initial number of clusters K, since all curves overlap. Therefore K is not a critical parameter, provided it is sufficiently larger than the number of sessions. In all the experiments, we choose $K = 1000$ for simplicity.

Fig. 5 shows instead the influence of the parameter g, that determines the value of the percentile used to select the cluster representatives in the cluster-to-cluster distance. We report i) the single linkage algorithm, which takes the two extreme values in the sample distribution as cluster representatives, ii) the centroid algorithm, which uses the mean value of the sample distribution, and iii) the percentile algorithm, which uses the gth percentiles, for variable values of $g \neq 0$. Regardless of the chosen distribution, the single linkage algorithm $(g = 0)$ has the best performance, while the centroid algorithm is the worst. Performance of the percentile algorithms improve as g decreases. The better accuracy of the single linkage algorithm is mainly due to the fact that gth percentile algorithms clusters real samples in the data set, and to the fairly large support of connection inter-arrivals, which calls for small values of g. Therefore, we opt for the single linkage algorithm in the remainder of the paper.

The choice of the single linkage algorithm solves also the issue of selecting a proper value for g, so that no parameters are required to run the algorithms. Indeed, the value of K must be determined at procedure startup. However, thanks to the preprocessing steps, the clustering algorithm operates only on TCP connections with the same client IP address in a relatively short time frame. Therefore, the initial number of clusters is neither related to the number of users accessing the Internet simultaneously, nor to the capacity of the access link. Simply upper bounding the number of sessions per user with a loose bound is enough to obtain accurate results, as previously shown. Considering a time interval of 10–12 hours, and given the average session duration, this bound can be easily obtained. As such, this parameter is easier to be determined with respect to the threshold in threshold based methodologies, which would instead require an *a priori* knowledge of user-session characteristics. Furthermore, the value of K is significantly less critical than the threshold value, as shown by the presented results.

B. Percentage of Misidentified Sessions

Let us now consider the percentage of sessions misidentified by the clustering procedure to assess the quality of the results and to compare them with those obtained via the traditional threshold based approach. Results reported in Fig. 6 show the percentage of errors obtained running the proposed clustering scheme and the threshold based scheme. The clustering scheme is run with an initial number of cluster $K = 1000$ and exploits the single linkage hierarchical agglomerative clustering $(g = 0)$. Performance of the classical threshold based scheme is evaluated, for variable values of the threshold η, from (2). A lower bound for the threshold-based algorithms, obtained using the optimal threshold value, is also shown. Exponential connection inter-arrivals are reported in the top plot, Pareto connection inter-arrivals in the bottom plot; clustering scheme performance is shown with a solid black line.

For all schemes, performance improves as $\overline{T}_{\text{off}}$ increases, since a longer silent period among user-sessions is more easily detected with respect to a short silent period among connection

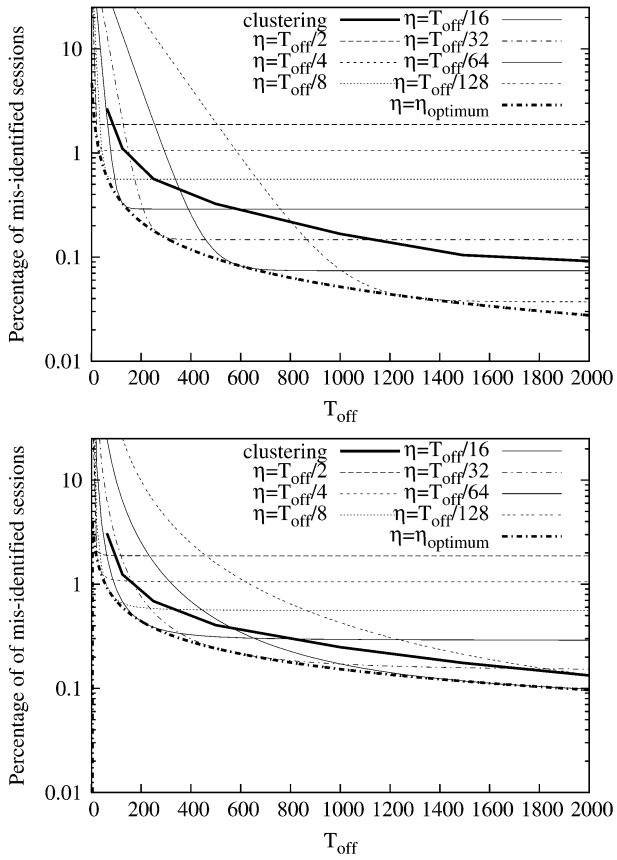

Fig. 6. Percentage of errors for exponential and Pareto connection inter-arrivals in top and bottom plots, respectively.

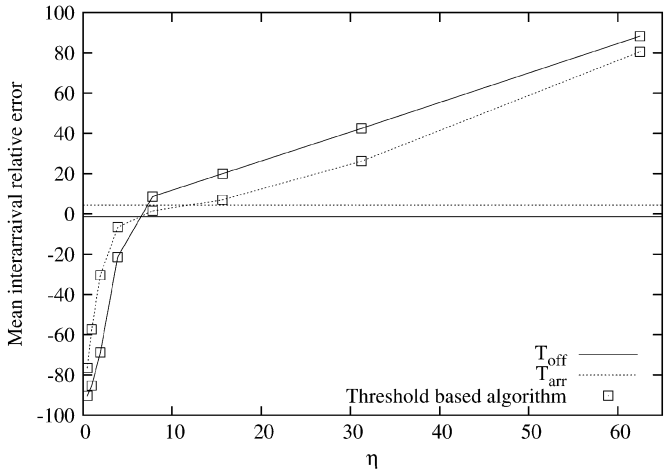

Fig. 7. Percentage of errors on the estimation of the mean OFF period (solid lines) and mean connection inter-arrival time (dotted lines). Clustering (lines) and threshold (lines with squares) algorithms as a function of the threshold η in threshold based algorithms.

arrivals. Threshold based mechanisms may perform better, provided that the proper threshold value is chosen. However, if the threshold is not correctly set, the error probability is much larger than the one shown by the clustering scheme. Large values of η induce a higher percentage of errors due to the probability of erroneously merging two subsequent different sessions. On the contrary, a sharp increase in error probability occurs for small values of η when $\overline{T}_{\rm off}$ goes below a given value, i.e., when $\overline{T}_{\rm off}$ becomes closer to $\overline{T}_{\rm arr}$.

When considering Pareto distributed connection inter-arrivals, no major differences are evident. For small values of $\overline{T}_{\rm off}$, an increase in the percentage of errors is observed for the threshold based approach. This is due to the higher probability that a connection inter-arrival time becomes comparable with the session OFF duration. The clustering approach is less affected by such events.

In summary, the clustering scheme shows a percentage of errors always smaller than 2%, and it is less sensitive to variations of $\overline{T}_{\rm off}$. Furthermore, it is more robust than the threshold based mechanism to variations of parameter settings.

C. Mean Inter-Arrival Estimation

To assess the accuracy of the clustering methodology in system parameters estimation, Fig. 7 reports the percentage of errors when estimating $\overline{T}_{\rm arr}$, the mean connection inter-arrival time, and $\overline{T}_{\rm off}$, the mean session OFF period duration. The clustering algorithm is plotted using lines, the threshold based approach, whose performance depends on the value assumed by η, using lines with squares. A rather hard scenario is considered, where connection inter-arrivals follow an exponential distribution with $\overline{T}_{\rm arr} = 1$ s, and $\overline{T}_{\rm off} = 62.5$ s. Errors are averaged over 5 runs, each run comprising 500 user-sessions.

The clustering approach is very accurate in the estimation of both $T_{\rm off}$ and $T_{\rm arr}$, with a relative error of about -1.4% in the estimation of $T_{\rm off}$, and 4.4% in the estimation of $T_{\rm arr}$. On the contrary, the threshold based approach is very sensitive to the choice of η. In particular, too many sessions are identified for $\eta < 5$, therefore underestimating both $T_{\rm arr}$ and $T_{\rm off}$. For $\eta > 10$, an overestimation is evident, due to the large number of erroneously merged sessions.

VII. Performance Analysis of Trace Data Set

The clustering methodology is now applied to measured data to determine Web user-session characteristics.

A. Web User-Session Characterization

Fig. 8 reports the CDFs of the mean connection inter-arrival $\overline{T}_{\rm arr}$ and of the mean user-session OFF period $\overline{T}_{\rm off}$. For each user, $\overline{T}_{\rm arr}$ and $\overline{T}_{\rm off}$ are first evaluated; then, the corresponding distributions over users are derived. Both October 02 and April 04 data set are considered. CDFs are similar, but $T_{\rm arr}$ assumes smaller values than $T_{\rm off}$. The two distributions overlap, as highlighted by the two vertical lines. This is due to the variability in user's behavior.

Note that no overlapping between the two distributions would have appeared if any threshold methodology had been applied, regardless of the adopted threshold. Furthermore, the variability of $\overline{T}_{\rm off}$ and $\overline{T}_{\rm arr}$ makes it very difficult to select an appropriate value for η. This highlights the drawbacks of threshold based approaches and the need of using clustering approaches that automatically adapt to different scenarios.

In Figs. 9, 10, 11, and 12 the main characteristics of Web user-sessions identified during April 04 are shown. PDFs are plotted using a linear/log scale, and the complementary CDF is shown using a log/log scale in the inset, to highlight the distribution tail.

Fig. 9 reports the PDF of the number of different server IP addresses in each session. Roughly 27% of sessions aggregate

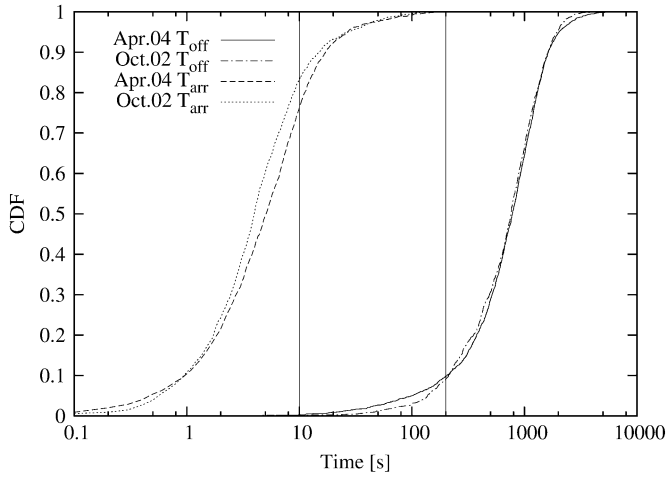

Fig. 8. Mean user-session and mean connection inter-arrivals CDFs.

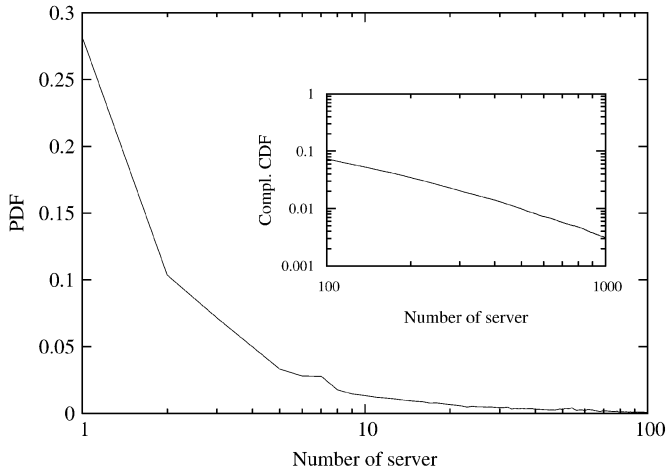

Fig. 9. PDF of the number of different server IP addresses per session. Complementary CDF in the inset.

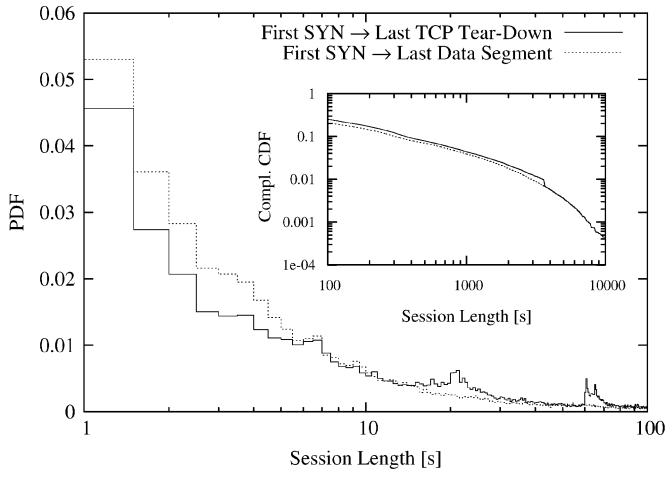

Fig. 10. PDF of session length. Complementary CDF in the inset.

connections from a single server, and about 10% of sessions refer to only two servers. However, the PDF has a heavy-tail, as highlighted by the complementary CDF, which shows that

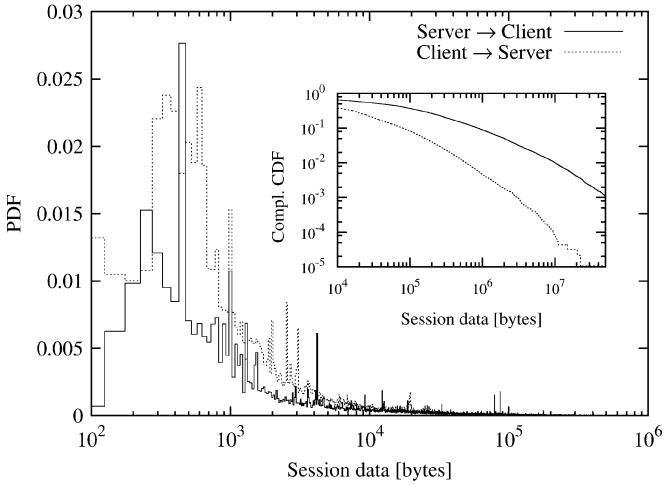

Fig. 11. PDF of the client-to-server and server-to-client data sent in each session. Complementary CDF in the inset.

the percentage of sessions contacting more than 100 different servers is not negligible.

Fig. 10 shows the session duration PDF. The two different distributions reflect the effect of different definitions of Web user-sessions. Indeed, user-session duration may be defined as: the time between the first SYN segment of the first connection and (i) the last segment observed during the last connection tear-down, for "protocol sessions"; (ii) the last segment carrying payload of the last connection for "application sessions". Therefore, using the notation introduced in Section IV, for a given session/cluster C, we can define the protocol session duration ΔT_e and the application session duration ΔT_d as

$$\Delta T_e = \max_{f_{\mathrm{id}} \in C}(t_e(f_{\mathrm{id}})) - \min_{f_{\mathrm{id}} \in C}(t(f_{\mathrm{id}}))$$
$$\Delta T_d = \max_{f_{\mathrm{id}} \in C}(t_d(f_{\mathrm{id}})) - \min_{f_{\mathrm{id}} \in C}(t(f_{\mathrm{id}})).$$

The protocol session definition is relevant, for example, when either Web servers or client resources are considered, since TCP connections must be managed until the tear-down procedure is completed. On the contrary, the application session definition is relevant for users, since users are satisfied when all data are correctly sent/received.

The protocol session distribution has obviously a larger support, but also biased peaks at 20 s, 60 s and 3600 s. They correspond to application layer timers imposed by Web browsers or HTTP servers which trigger the connection tear-down procedure after idle periods. For example, Web servers may wait for a timer to expire (usually after 20 seconds) before closing the connection. Similarly, HTTP 1.1 and Persistent-HTTP 1.0 protocols use an additional timer, usually set to a multiple of 60 seconds. Therefore, the protocol session duration highlights the bias induced by those timers. The bias disappears when application session duration is evaluated.

Session duration distributions have a large support, showing large variability in user's behavior. Indeed, there is a large percentage of very short sessions (that last less than few seconds), but also user activities that last for several hours. The tail of the complementary CDF shown in the inset highlights the heavy-tailed distribution of session duration.

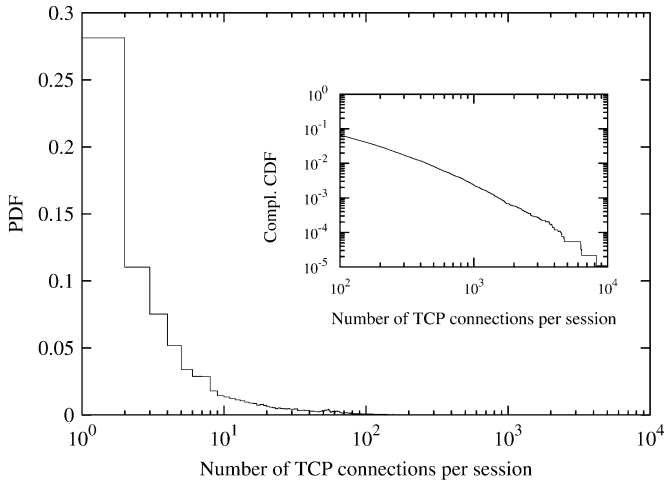

Fig. 12. PDF of the number of TCP connection in each session. Complementary CDF in the inset.

Fig. 11 reports the PDF of the amount of data exchanged from client to server D_c (dashed lines) and server to client D_s (solid lines). For a given session/cluster $C, D_c = \sum_{f_{\mathrm{id}} \in C}(B_c(f_{\mathrm{id}}))$ and $D_s = \sum_{f_{\mathrm{id}} \in C}(B_s(f_{\mathrm{id}}))$. As expected, more data are transferred from servers to clients, and the distribution tail is heavier; the number of sessions transferring more than 10 Mbytes in the server-client direction is not negligible. The initial part of both PDFs presents a number of peaks. Investigating further, we discovered that peaks are due to the identification of sessions which are not generated by users, but instead by automatic reload procedure imposed by the Web page being displayed. For example, news or trading on-line services impose periodic updates of pages, which force the client to automatically reload the pages. If the automatic reload is triggered periodically, the clustering algorithm may identify a separate session for each connection, thus causing a bias in the session data distribution.

This is clearly evident also from Fig. 12, which reports the number of TCP connections per user-session. Indeed, more than 25% of sessions include only one TCP connection. Furthermore, most of the identified sessions are composed by very few connections (about 50% by 4 connections or less). This demonstrates that: i) the client is usually able to obtain all the required data using few TCP connections, ii) the number of required external objects is limited, and iii) the time spent by the users over one Web page is large enough to define each Web transaction as a session. The CDF, reported in the inset, shows a linear trend, highlighting that the distribution has a heavy-tail.

B. Session Inter-Arrivals Statistical Properties

Finally, statistical properties of session inter-arrival times are investigated. A session arrival trace is obtained by superimposing in time all identified sessions during the same time period.

Fig. 13 reports the Q-Q plot of the session inter-arrival distribution with respect to the best fitted Weibull distribution over the same data set. The choice of the Weibull model stems from the fact that connection arrivals fit quite well a Weibull distribution with a heavy-tail [26]. The Weibull distribution is characterized by the so called "shape" and "scale" parameters. When the

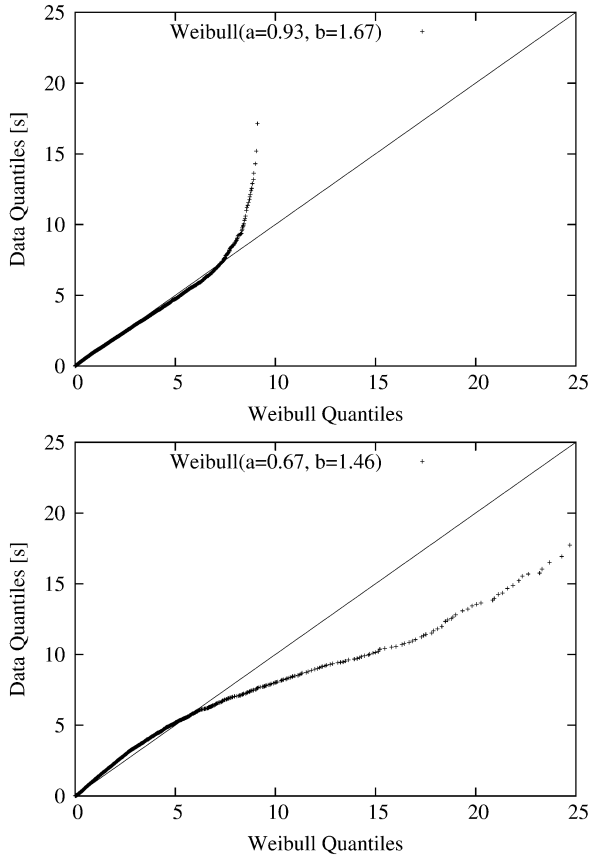

Fig. 13. Fit of user-session inter-arrivals to a Weibull distribution: normal working day in the top plot, and during a worm attack on the bottom plot.

shape parameter, named a in this paper, is set to 1, the Weibull distribution degenerates into an exponential distribution. When a is smaller than 1, the tail of the distribution is heavy, while for values of a larger than 1 the shape of the distribution assumes a dumbbell form. The classical maximum likelihood method was used to obtain the parameters a and b for the fitting procedure. We also used the Anderson–Darling (A-D) test [27] to test the fitting quality under the null hypothesis that inter-arrivals are drawn from a Weibull distribution whose parameters are taken from the maximum likelihood estimations.

The top plot of Fig. 13 refers to a typical measurement day and shows a good matching of data samples with the chosen Weibull distribution. Being $a = 0.93$, it also shows that the distribution is very close to an exponential distribution, therefore hinting that the arrival process of Web user-session is Poisson, as pointed out in previous studies [7]. The Q-Q plot shows also that the tail of the distribution is less heavy than the tail of the fitted Weibull distribution. Therefore, there is a bias toward small values of session inter-arrivals, with rare large inter-arrivals. The Q-Q plot of the same data for the best-fitted exponential distribution showed almost the same behavior, and is not reported for the sake of conciseness.

On the contrary, the bottom plot in Fig. 13, which refers to samples collected on May 5th, 2004, shows a distribution that is not well fitted by a Weibull distribution. The best fit is obtained by a shape parameter $a = 0.67$, indicating a heavy-tailed distribution. Moreover, the best fit Weibull distribution deviates

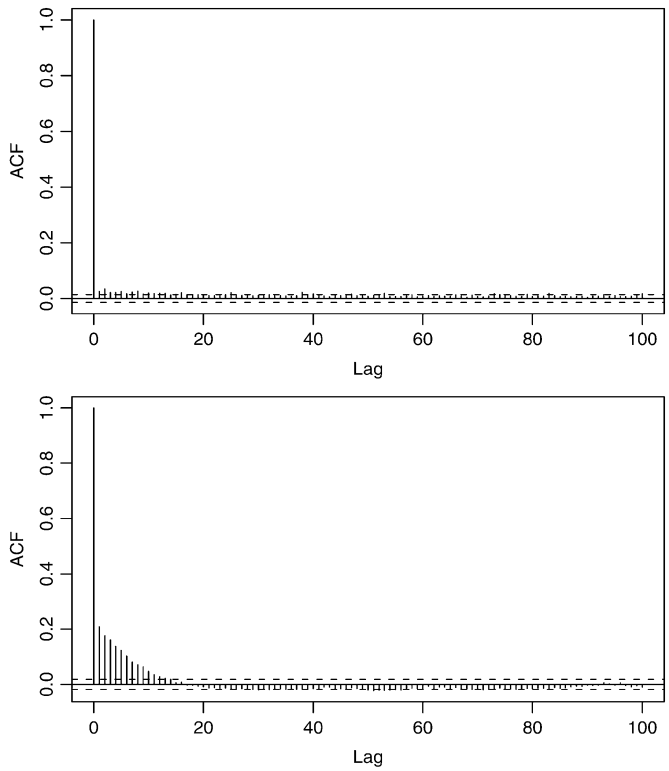

Fig. 14. Auto correlation function of the session inter-arrival process: normal day in the top plot, during a worm attack on the bottom plot.

from the measured data set on large quantiles, showing that the tail of the real distribution is heavier than the one of the best fit distribution. The anomalous behavior is due to the spreading of the Sasser.B worm [23] in our LAN institution, which caused several issues to the hosts and to the network. Indeed, users were forced to download operating system patches and antivirus updates. Furthermore, our campus network and the Internet in general suffered from problems related to the worm spreading, e.g., forced shutdowns of entire subnets, sudden congestion on links and firewalls, etc. Both events introduced correlation at the session level, created by large file download times and degraded network performance. This correlation is reflected by the session arrival process, which deviates significantly from the Poisson assumption.

To make sure that the anomalies were not introduced by our methodology, we also tried to fit the session inter-arrival distribution as identified by a threshold procedure. The qualitative results are similar, even if the quantitative measurements (e.g., mean inter-arrival time) are obviously different and strongly depending on the selected threshold.

Finally, Fig. 14 reports the autocorrelation function evaluated on session inter-arrivals during a typical day on the top plot, while the bottom plot refers to the autocorrelation estimated during the day of the worm attack. The top plot confirms that the Poisson assumption holds for normal days, being the autocorrelation function almost negligible except in the origin. On the contrary, as shown in the bottom plot, the autocorrelation function is quite relevant on days during which user activities are driven by external factors such as worm infection.

VIII. Conclusion

Clustering techniques were applied to a large set of real Internet traffic traces to identify Web user-sessions. A novel clustering methodology was proposed and compared with the classical threshold based scheme.

The effectiveness and robustness of the proposed clustering methodology was first assessed by applying it to an artificial data set, and showing its ability in the identification of Web user-sessions without requiring any *a priori* definition of threshold values. Then, the proposed clustering methodology was applied to measured data sets to study the characteristics of Web user-sessions. User-sessions were shown to be Poisson. However, correlation arises when an anomalous network behavior is induced, for example, by a worm infection. The analysis of the identified user-sessions shows a wide range of diverse behaviors that cannot be captured by any threshold based scheme.

The clustering algorithm proposed in this paper can be helpful in studying traffic properties at the user level, and could be easily extended to deal with other types of user-sessions, not necessarily related to Web traffic.

References

[1] V. Paxson and S. Floyd, "Wide-area traffic: The failure of Poisson modeling," *IEEE/ACM Trans. Netw.*, vol. 3, no. 3, pp. 226–244, Jun. 1995.

[2] M. E. Crovella and A. Bestavros, "Self similarity in world wide web traffic: Evidence and possible causes," *IEEE/ACM Trans. Netw.*, vol. 5, no. 6, pp. 835–846, Dec. 1997.

[3] R. Caceres, P. Danzig, S. Jamin, and D. Mitzel, "Characteristics of wide-area TCP/IP conversations," in *Proc. ACM SIGCOMM'91*, Aug. 1991, pp. 101–112.

[4] P. Danzig and S. Jamin, "Tcplib: A library of TCP Internetwork traffic characteristics," USC, Tech. rep., 1991.

[5] P. Danzig, S. Jamin, R. Caceres, D. Mitzel, and D. Mestrin, "An empirical workload model for driving wide-area TCP/IP network simulations," *Internetworking: Research and Experience*, vol. 3, no. 1, pp. 1–26, 1992.

[6] V. Paxson, "Empirically derived analytic models of wide-area TCP connections," *IEEE/ACM Trans. Netw.*, vol. 2, no. 4, pp. 316–336, Aug. 1994.

[7] C. Nuzman, I. Saniee, W. Sweldens, and A. Weiss, "A compound model for TCP connection arrivals, with applications to LAN and WAN," *Computer Networks, Special Issue on Long-Range Dependent Traffic*, vol. 40, no. 3, pp. 319–337, Oct. 2002.

[8] F. D. Smith, F. H. Campos, K. Jeffay, and D. Ott, "What TCP/IP protocol headers can tell us about the web," *SIGMETRICS Perform. Eval. Rev.*, vol. 29, no. 1, pp. 245–256, 2001.

[9] W. Willinger, V. Paxson, and M. S. Taqqu, "Self-similarity and heavy-tails: Structural modeling of network traffic," in *A Practical Guide to Heavy Tails: Statistical Techniques and Applications*, R. Adler, R. Feldman, and M. S. Taqqu, Eds. Boston, MA: Birkhauser, 1998.

[10] T. Bonald, A. Proutière, G. Régnié, and J. W. Roberts, "Insensitivity results in statistical bandwidth sharing," in *Proc. Int. Teletraffic Congr. (ITC) 17*, Salvador, Brazil, Dec. 2001, 12 pp.

[11] P. Barford and M. Crovella, "Generating representative web workloads for network and server performance evaluation," in *Proc. SIGMETRICS'98/PERFORMANCE'98*, 1998, pp. 151–160.

[12] L. Cherkasova and P. Phaal, "Session-based admission control: A mechanism for peak load management of commercial Web sites," *IEEE Trans. Comput.*, vol. 51, no. 6, pp. 669–685, Jun. 2002.

[13] M. Pioro and D. Medhi, "Routing, flow, and capacity design in communication and computer networks," *The Morgan Kaufmann Series in Networking*, 2004.

[14] *Universal Mobile Telecommunications System (UMTS), Selection Procedures for the Choice of Radio Transmission Technologies of the UMTS (UMTS 30.03 Version 3.2.0)*, ETSI TR 101 112 V3.2.0 (1998-04), (1998-04).

[15] S. Ben Fredj, T. Bonald, A. Proutière, G. Régnié, and J. W. Roberts, "Statistical bandwidth sharing: A study of congestion at flow level," in *Proc. ACM SIGCOMM 2001*, San Diego, CA, Aug. 2001, pp. 111–122.

[16] M. F. Arlitt and C. L. Williamson, "Web server workload characterization: The search for invariants," in *Proc. ACM SIGMETRICS'96*, Philadelphia, PA, 1996, pp. 126–137.
[17] Y. Fu, A. Vahdat, L. Cherkasova, and W. Tang, "EtE: Passive end-to-end Internet service performance monitoring," in *Proc. General Track: 2002 USENIX Annu. Tech. Conf.*, Monterey, CA, 2002, pp. 115–130.
[18] D. J. Hand, H. Mannila, and P. Smyth, *Principles of Data Mining*. Cambridge, MA: MIT Press, 2001.
[19] A. Bianco, G. Mardente, M. Mellia, M. Munafò, and L. Muscariello, "Web user session characterization via clustering techniques," in *Proc. IEEE GLOBECOM 2005*, St. Louis, MO, Nov. 2005, vol. 2, pp. 1102–1107.
[20] The GARR Network Topology. 2005 [Online]. Available: http://www.garr.it/reteGARR/mappe.php
[21] M. Mellia, A. Carpani, and R. Lo Cigno, "Measuring IP and TCP behavior on edge nodes," in *Proc. IEEE GLOBECOM 2002*, Taipei, Taiwan, R.O.C., Nov. 2002, vol. 3, pp. 2533–2537.
[22] M. Mellia, R. Lo Cigno, and F. Neri, Tstat Web Page. 2001 [Online]. Available: http://tstat.tlc.polito.it
[23] *What You Should Know About the Sasser Worm*. May 2004 [Online]. Available: http://www.microsoft.com/security/incident/sasser.mspx
[24] S. McCanne, C. Leres, and V. Jacobson, Tcpdump. 2001 [Online]. Available: http://www.tcpdump.org
[25] G. Iannaccone, C. Diot, I. Graham, and N. McKeown, "Monitoring very high speed links," in *Proc. ACM SIGCOMM Internet Measurement Workshop*, San Francisco, CA, Nov. 2001, 5 pp.
[26] A. Feldmann, "Characteristics of TCP connection arrivals," AT&T Labs Research, Florham Park, NJ, Tech. rep. , 1998.
[27] *Goodness-of-Fit Techniques*, R. B. D'Agostino and M. A. Stephens, Eds. New York: Marcel Dekker, 1986.

Andrea Bianco (M'97) is Associate Professor in the Electronics Department of Politecnico di Torino, Italy. He has co-authored over 100 papers published in international journals and presented in leading international conferences in the area of telecommunication networks. His current research interests are in the fields of protocols and architectures of all-optical networks and switch architectures for high-speed networks.

Dr. Bianco was technical program co-chair of HPSR 2003 and 2008 and DRCN 2005. He has been a TPC member of several conferences, including IEEE INFOCOM, IEEE GLOBECOM, IEEE ICC, and HPSR.

Gianluca Mardente received the degree in electronic engineering and the Ph.D. degree in telecommunications engineering from Politecnico di Torino, Italy, in 2001 and 2005, respectively.

He is a Senior Software Engineer with Cisco Systems, San Jose, CA. His former research interests include routing algorithms, QoS in IP networks, and traffic measurement, modeling and characterization. He is currently involved in the design of system performance analysis of next generation routers.

Marco Mellia (M'97) is an Assistant Professor in the Electronics Department of Politecnico di Torino, Italy. He has co-authored over 80 papers published in international journals and presented in leading international conferences, all of them in the area of telecommunication networks. His research interests are in the fields of traffic classification, measurement and modeling, and of peer to peer systems.

Dr. Mellia has been TPC member of several conferences, including IEEE INFOCOM, IEEE GLOBECOM and IEEE ICC.

Maurizio M. Munafò (M'98) received the Dr. Ing. degree in electronic engineering and the Ph.D. degree in telecommunications engineering from Politecnico di Torino, Italy, in 1991 and 1994, respectively.

He is an Assistant Professor in the Electronics Department of Politecnico di Torino. He has co-authored about 60 journal and conference papers in the area of communication networks and systems. His current research interests are in simulation and performance analysis of communication systems, traffic modeling and identification of worms and traffic anomalies.

Luca Muscariello (M'01) graduated in telecommunications engineering from Politecnico di Torino, Italy, in 2002. He received the Ph.D. degree in telecommunications engineering from Politecnico di Torino, Italy, labeled as "European Doctorate" by the Conference of European Rectors. His graduate research was performed in the area of Internet traffic measurement, characterization and modeling.

He is an R&D Engineer at France Telecom R&D, Paris, France. During 2006, he was a postdoctoral researcher in France Telecom R&D, working on performance evaluation of wireless networks. His current research interests include simulation and analytical modeling of wired and wireless networks.